

Installing NaturalX

This section describes how to install NaturalX and covers the following topics:

- Installing NaturalX under Windows 98/NT/2000
 - Installing NaturalX under UNIX
 - NaturalX System Architecture under OS/390
 - Installing NaturalX under OS/390
-

Installing NaturalX under Windows 98/NT/2000

NaturalX is part of Natural for Windows 98/NT/2000 and is automatically installed with the installation of Natural for Windows 98/NT/2000.

Installing NaturalX under UNIX

This section describes how to install NaturalX under UNIX.

Prerequisites

Before you begin to install NaturalX under UNIX, ensure that your computer meets the following prerequisites:

- EntireX DCOM Version 5.3.1
- For access from Windows NT to UNIX:
Windows NT Version 4.0 with Service Pack 6.
- Natural 5.1.1 for UNIX

Environment Variables

Make sure that the environment variables for EntireX DCOM have been set according to the EntireX DCOM documentation.

In order to start up correctly, Natural needs the environment variables NATDIR and NATVERS. To make sure that these environment variables are also set when a NaturalX server is launched automatically by EntireX DCOM, NATDIR and NATVERS must also be set in the environment where the ntd daemon (EntireX DCOM) is started. Set NATDIR to the path where Natural is installed and NATVERS to the name of the Natural version directory.

Installation Procedure

Step 1 - Perform the General Installation Procedure for Software AG Products for UNIX

For information on this subject, read Installing And Setting Up Software AG Products for UNIX in the Natural Installation and Operations Manual for OpenVMS and UNIX.

This section contains general information which applies when installing and setting up any Software AG product on a UNIX platform.

Step 2 - Execute the NaturalX Installation Script

To install NaturalX, you use the installation script "nxxinstall.bsh".

1. Issue the following commands to execute the installation script:

```
cd $NXXDIR/$NXXVERS/INSTALL.  
/nxxinstall.bsh
```

2. Follow the instructions provided by the installation script.
A new Natural nucleus with NaturalX support will be linked.
A backup copy of the file "natural" will be created and named "natural.old".

NaturalX System Architecture under OS/390

Before you start to install NaturalX under OS/390, it is important that you have a clear picture of the NaturalX system architecture under this operating system.

This section covers the following topics:

- Overview
- Environment Variables
- NaturalX Server Front-End
- NaturalX Output Files
- The NaturalX Server Monitor
- The DCOM Buffer Pool

Overview

NaturalX is available for OS/390 (Version 2.4 or above) and requires EntireX DCOM Version 5.2.1 or above. It uses the OS/390 UNIX Services and must have access to the OS/390 HFS (hierarchical file system). Usability is restricted to Natural sessions running in batch-oriented systems (TSO, batch and processes in OS/390 UNIX Services).

NaturalX consists of the following four load modules:

- NATCOMST resides in a PDS load library. This module is the root entry point for the Natural COM function and is loaded by the Natural nucleus at session initialization. It provides the nucleus with a list of the entry points for the APIs in NATCOM.
- NATCOM resides in a PDS load library. This module contains the API functionality which enables the Natural nucleus to call functions in EntireX DCOM.
- The load module "naturalx" resides in the HFS and is used by EntireX DCOM to start a NaturalX DCOM server. NaturalX is a multi-threading application which is able to start and execute multiple Natural sessions on multiple clients simultaneously. The Natural sessions are executed within a configurable number of storage threads and the session data is swapped either using the Natural Roll Server or within the virtual storage. Swapping using virtual storage indeed limits the number of parallel sessions.
- The load module "natxmon" resides in the HFS and is used to monitor NaturalX server processes. For example, NATXMON can be used to terminate a specific NaturalX server process manually.

NaturalX Server

A NaturalX server is a process which is responsible for executing a Natural session which hosts classes, for further information, see the section Distributing Object-Based Natural Applications. Because mainframe Natural is a threading system, a NaturalX server on OS/390 is designed as a multi-user application able to maintain multiple Natural sessions for different clients. A class registered with activation policy *ExternalSingle* for example does not have to be hosted by an exclusive process, as it is hosted by an exclusive Natural session.

For sessions hosted by one server process, the following resources are exclusive:

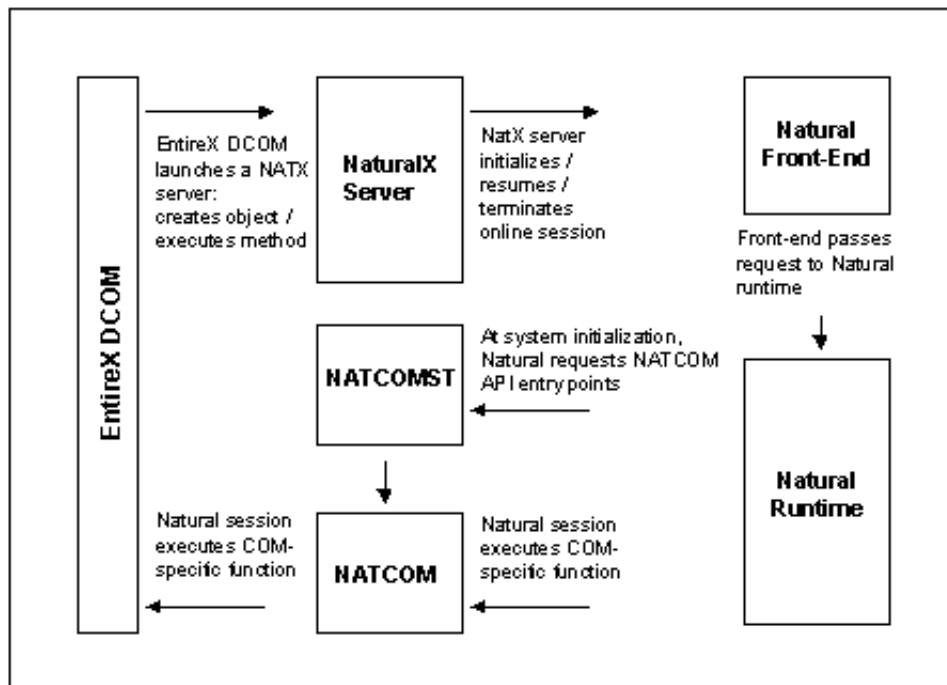
- Each session has its own database user ID,
- Each session has its own Natural session data (e.g. system variables, DATSIZE, error stack)

For sessions hosted by one server process, the following resources are common:

- All sessions run within one OS/390 address space.
- All sessions are started with the same session parameters.
- All DB2 access calls are made under the user ID of the server process.
- Print and work files can either be shared between sessions or be used exclusively by one session. For a detailed description, see the section **Natural as a Server** under **Natural in Batch Mode** in the section **Environment-Specific Information** in the Natural Operations for Mainframes documentation.

The following diagram gives an overview of the components involved in a NaturalX server environment:

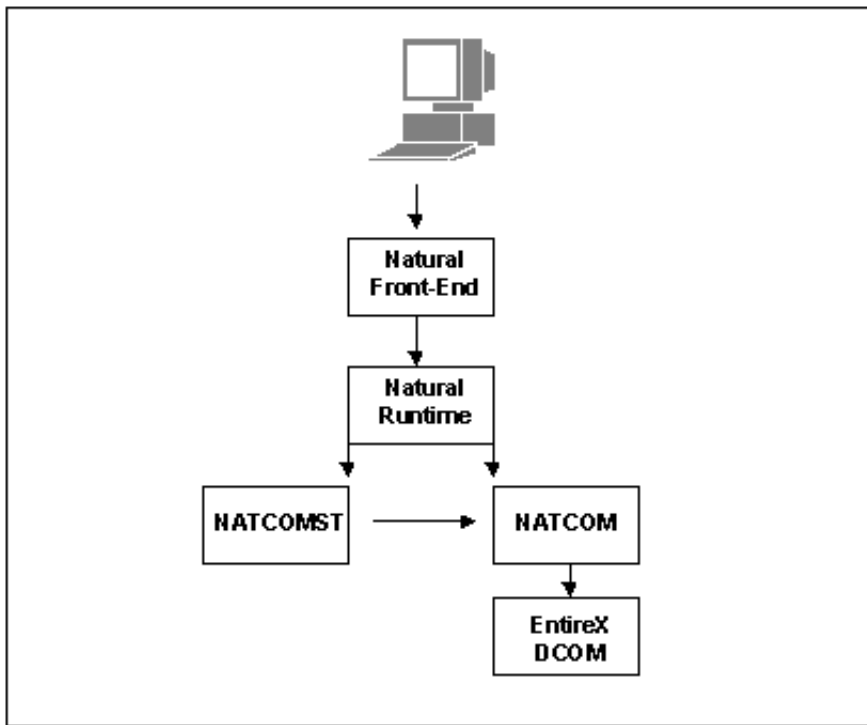
NaturalX Components in a Server Environment



NaturalX Client

A NaturalX COM client can be any Natural TSO or batch session which uses distributed COM objects. The following diagram gives an overview of the components involved in a NaturalX client session:

NaturalX Components in a Client Environment



Environment Variables

The execution of NaturalX is configured using UNIX environment variables. Some of these variables are required for COM clients, some for COM servers and some for both. The variables mentioned in the EntireX DCOM documentation must always be set. How the variables are set depends on how you start your Natural online session or your NaturalX COM server.

Starting Natural under TSO or Batch

If you start a Natural session in TSO or batch, an HFS file must be allocated to the DD name NATXENV. This HFS file contains the variable assignments.

Example of an HFS File Allocated to NATXENV

```
PATH=/u/dcomkit/SAG/dco/v411/bin:/u/nat/v311/bin
LIBPATH=/u/dcomkit/SAG/dco/v411/lib
_CEE_RUNOPTS=TERMTHDACT(MSG),POSIX(ON),STACK(4M,1M,ANY,KEEP),STORAGE(NONE,NONE,
NONE,8K),HEAP(32K,32K,ANYWHERE,KEEP,8K,4K)
SAGNODE=DAEY
NATDIR=/u/SAG/natural
NATVERS=v311
NATX_NUCNAME=NATOS31L
NATX_NTHREADS=2
NATX_THREADSIZE=300
NATX_TRACE=1
```

Example for TSO ISPF procedure

```
PROC 0
  CONTROL NOFLUSH ASIS LIST CONLIST
  ALLOC FILE(NATXENV) +
    PATH(' /u/SAG/NAT/V311/natxenv' ) PATHOPTS(ORDONLY)
  ALLOC FILE(CMPRINT) DA(*)
  ALLOC FILE(CMSYNIN) DA(*)
  ALLOC FILE(SYSOUT) DA(*)
  CALL 'SAG.NAT311.LOAD(NAT31)'
```

Starting NaturalX Servers Manually

Usually EntireX DCOM launches the NaturalX server automatically if any client requests a server which is currently not active. It is always possible to start a NaturalX COM server manually under the OS/390 UNIX shell by executing the executable "naturalx". The variables must be defined in the shell environment from which you start NaturalX. NaturalX requires the session parameter COMSERVERID=*serverid* (for OS/390, DCOM=(SERVID=*serverid*)).

Example

```
naturalx "DCOM=(SERVID=MYSERVER)"
```

It is not possible to start a conventional Natural session as a COM server. A NaturalX COM server must be started by the executable "naturalx".

Starting NaturalX Servers from EntireX DCOM

If EntireX DCOM (RPCSS) launches the NaturalX COM server, NaturalX inherits the environment variables from EntireX DCOM. All the environment variables required by NaturalX must be defined in the shell environment from which EntireX DCOM is started.

List of NaturalX Environment Variables

In the sections below, the characters in brackets indicate whether a variable is required for clients (C), servers (S) or both (C/S):

DCOLIB - C

Points to the directory in which the standard OLE type library "stdole32.tlb" is stored. "stdole32.tlb" is included in the type library which is created when a class is registered.

Example for EntireX DCOM 5.2.2

```
DCOLIB= "/EXXDIR/EXXVERS/lib"
```

NATDIR - C/S

Points to the Natural HFS root directory.

Example

```
NATDIR=/u/SAG/natural/
```

NATVERS - C/S

Specifies the version-dependent subdirectory of NATDIR.

Example

```
NATVERS=v311
```

NATX_DELAY

Usually a COM server terminates if the number of objects hosted is zero. NATX_DELAY is used to delay server termination.

Examples

```
NATX_DELAY=30S causes termination delay of 30 seconds
NATX_DELAY=10M causes termination delay of 10 minutes
NATX_DELAY=1H causes termination delay of 1 hour
NATX_DELAY=INFINITE causes no termination
```

NATX_DYNALLOC

$$\text{NATX_DYNAL LOC} = \left\{ dd\text{-}name, \left\{ \begin{array}{l} \text{HFS} = \text{hfs-file}name \\ \text{PDS} = \text{dataset-na}me \\ \text{SYSOUT} = \text{output-class} \end{array} \right\} \right\} : \dots$$

Specifies DD name allocations for NaturalX servers by defining a list of DD names which are allocated to datasets.

dd-name	DD name with max. 8 characters (for example SYSUDUMP)
hfs-filename	hfs file name with absolute path definition
dataset-name	existing and catalogued dataset name
output-class	single character JES output class

Example

```
NATX_DYNALLOC="CMPRINT,HFS=/u/tmp/myfile:SYSUDUMP,PDS=NAT.DUMP.F01:CMPT01,SYSOUT=X"
```

NATX_DYNALLOC allocates the DD name

- CMPRINT to HFS file 'myfile' on directory /u/tmp
- SYSUDUMP to dataset NAT.DUMP.F01
- CMPT01 to output class X.

NATX_FEOPT - S

Specifies additional options for the Natural front-end as follows:

01	Do not use the roll server
02	Clean up roll file at server termination

Example

```
NATX_FEOPT=01
```

NATX_FEPRM

Specifies additional Natural Front-End parameters as specified in the Startup Parameter Area. You can define multiple parameters. Each parameter is specified by a pair of 8-character strings of which the first contains the parameter keyword and the second, the parameter value. For further information, see the Natural Operations for Mainframe documentation, section **Environment-Specific Information**, section **Natural in Batch Mode**.

Example

```
NATX_FEPRM="MSGCLASSX"
```

This setting determines that the default output class for CMPRINT is "X".

NATX_INITTOUT

Specifies the number of seconds the client must wait until the launched NaturalX server is initialized. The default value is 10.

Example

```
NATX_INITTOUT=5
```

The client waits at most 5 seconds until the server is initialized.

The error message NAT0711 with the DCOM code 8004100C occurs if the timeout limit is reached.

NATX_NTHREADS - S

The number of physical storage threads to be allocated by the Natural front-end. The number of sessions which can be executed in parallel.

Note:

This number does not limit the number of sessions within the server, but the number of sessions which can be in execution status concurrently. The number of sessions is limited by the size of the Natural swap medium.

Example

```
NATX_NTHREADS=5
```

NATX_NUCNAME - S

The name of the Natural front-end to start a Natural session. The front-end resides on a PDS member. For further information, see the section NaturalX Server Frontend.

Example

```
NATX_NUCNAME=NAT311SV
```

NATX_THREADSIZE - S

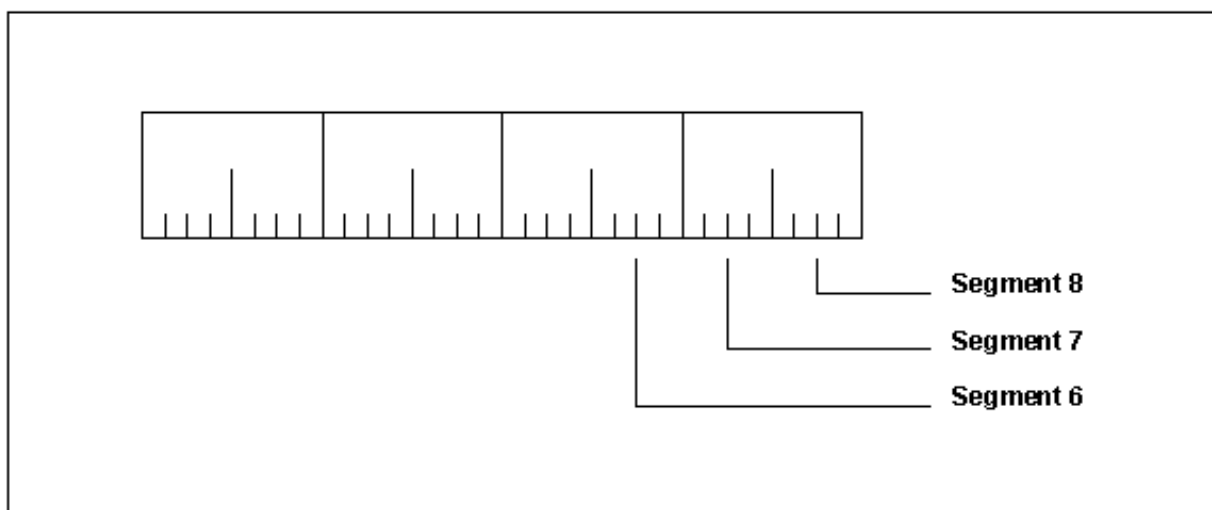
The size (in KB) of each physical storage thread which contains the Natural session data at execution time.

Example

```
NATX_THREADSIZE=800
```

NATX_TRACE - S

Defines the trace level of the server. The value is used as an 8 X 4-bit flag container in which each 4-bit segment controls the trace of a particular NaturalX functional unit.



Segment	Description
1 - 5	Not used.
6	Controls the trace output of NATCOM client calls.
7	Controls the trace output of server calls
8	Controls the trace output of the NaturalX front-end stub.

A segment consists of four bits where the first bit is currently not used. The value of a segment can thus be in the range 0 - 7. The higher the value, the more extensive the trace output.

Example

```
NATX_TRACE=0x00000172
```

This setting determines a level 1 trace for client calls, a level 7 trace for server calls and a level 2 trace for the NaturalX font-end stub.

PATH - C/S

Refers to the DCOM and Natural bin directory. For further information, see your OS/390 documentation.

Example

```
PATH=/u/SAG/dco/bin:u/SAG/nat/bin
```

STEPLIB - S

Refers to the PDS where the Natural front-end is installed. For further information, see your OS/390 documentation.

Example

```
STEPLIB="RZ.NAT311.LOAD"
```

Note:

The PDS load libraries defined in the STEPLIB environment variable must be defined in the 'system sanction list for set-user-id and set-group-id programs'. The 'system sanction list' is a HFS file containing the dataset names available for processes which are invoked under a different user-ID. For more information on the 'system sanction list', see the description of the statement STEPLIBLIST in the IBM manual OS/390 V2R4.0 OpenEdition Planning.

NaturalX Server Front-End

The NaturalX server front-end is a standard MVS batch driver which is assembled with the NTOS parameter LE370=POSIX. The same applies to the TSO driver if you execute COM requests within your TSO Natural online session.

NaturalX Output Files

For tracing purposes NaturalX allocates three output files for each COM server ID and three output files for each client user ID. The files can be found under:

```
NATDIR/NATVERS/trace/server/serverid.trc  the stub trace file
                                /serverid.sto  the stub stdout file
                                /serverid.ste  the stub stderr file
```

and

```
NATDIR/NATVERS/trace/client/userid.trc    the stub trace file
                                /userid.sto   the stub stdout file
                                /userid.ste   the stub stderr file
```

The stdout and stderr files for clients do not exist if the client session is hosted by a COM server (that is, a COM server session becomes an agent). In that case the output of stderr and stdout is directed to the appropriate server file. The output files are always removed at server/client initialization.

The NaturalX Server Monitor

The server monitor "natxmon" can be used to send messages to a specific NaturalX server process via standard UNIX message queues in order to administer a server process without using COM. "natxmon" is used from the OS/390 UNIX shell with the following two parameters:

natxmon pid message

Parameter	Description
<i>pid</i>	Contains the process ID of the server process.
<i>message</i>	Contains one of the following numeric values: <ol style="list-style-type: none"> 1. The server writes status information to the trace file. 2. The server deregisters and terminates. 3. The server aborts.

Example

```
natxmon 4711 2
```

This setting terminates the NaturalX server with the PID 4711.

The DCOM Buffer Pool

The COM implementation is based on VTABLES (virtual function tables) which contain lists of entry points which are used to call the object methods. A VTABLE is not relocatable because its address itself identifies a specific class object. Natural collates all VTABLES within a DCOM buffer pool. The allocation is similar to already existing Natural buffer pools, for example, the Sort and Editor buffer pools. A new buffer pool type DCOM has been introduced. For further information, see the BPI parameter documentation. The buffer pool type DCOM can be either local or global.

Installing NaturalX under OS/390

Installing NaturalX under OS/390 is described in the Natural Installation Guide for Mainframes.